

APPLYING LINEAR ALGEBRA TO IMAGE DEBLURRING

By

John Julian Sanborn

A thesis submitted to the University Honors Program at Southern New Hampshire University to complete HON 401, and as part of the requirements for graduation from the University Honors Program

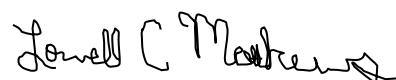
Manchester, New Hampshire

May 2019

Approved by:



(Faculty Mentor)



University Honors Director

John Sanborn

Dr. William Jamieson

Applying Linear Algebra to Image Deblurring

Abstract Intro

This thesis expands on the concepts taught in Applied Linear Algebra (MAT-350), using singular value decomposition (SVD) and discrete cosine transform (DCT), with a focus on image deblurring. The principles discussed throughout this thesis were guided by the readings of *Deblurring Images Matrices, Spectra, and Filtering* by Christian Hansen, James Nagy, and Dianne O'Leary. The thesis will focus on various techniques that were used to deblur an image, how the SVD and DCT were applied, and the results applied to a blurred photo. The mathematics are made easier using MATLAB's built-in tools including: The *Signal Processing Toolbox* (SPT) and the *Image Processing Toolbox* (IPT) as well as tools created by the authors of *Deblurring Images Matrices, Spectra, and Filtering*. The goal of this project is to not only learn the theoretical side of the mathematics behind image deblurring, but also to write code to implement various techniques used to deblur an image.

Keywords: *Singular Value Decomposition, Discrete Cosine Transform, Matrix Decomposition, Spectral Filtering*

Introduction

Consider an image that is captured by a digital camera. The image is transformed into a matrix with integer values using mechanical sensors which inherently introduce error into the signal, called discretization error. Human error can also be introduced to the signal if the camera operator does not properly focus the camera or if the camera is moving while a picture is being taken. To fix these errors, one may apply linear algebra concepts.

According to [1], a digital image consists of pixels which are assigned integer values to represent a certain color intensity. Depending on the size and resolution of the photo, a small image may have 65,536 pixels (256^2) whereas a high-resolution image could have anywhere between 5 to 10 million pixels. Image deblurring/image restoration is the process of removing the blur within a photo. The blurring occurs because of pixel intensities being averaged over some area. To deblur a photo, we can use a mathematical process based on concepts of linear algebra.

Matrix

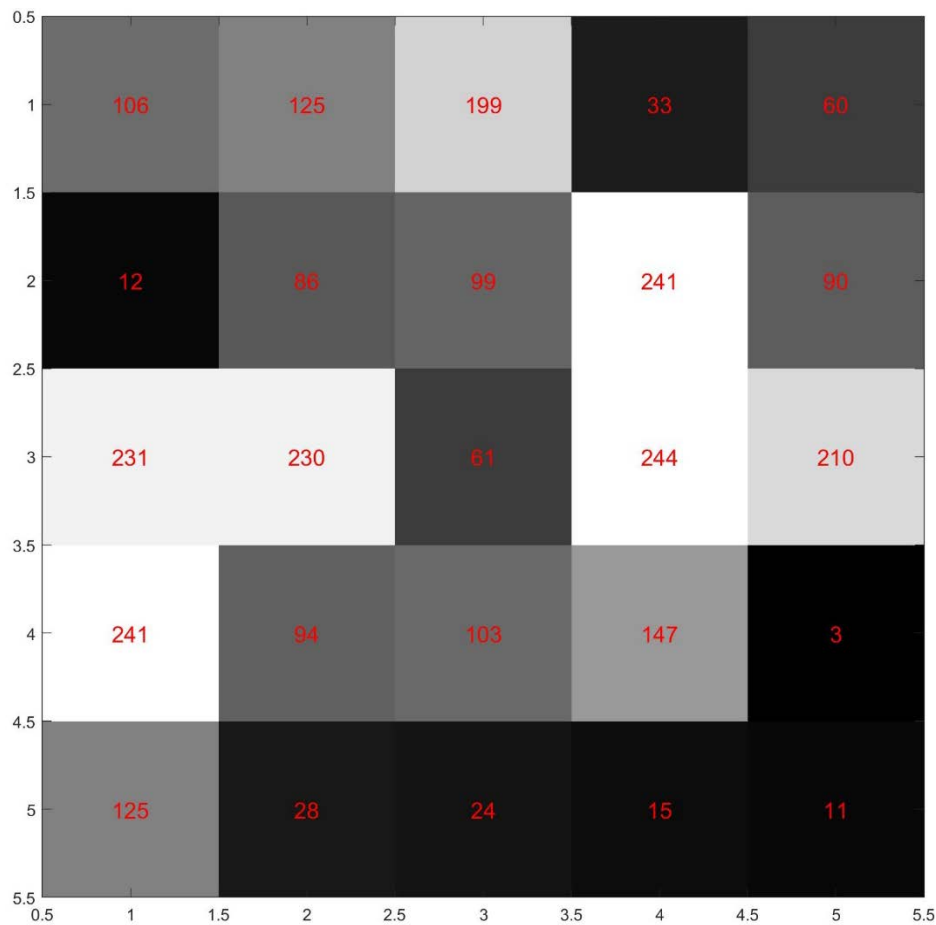


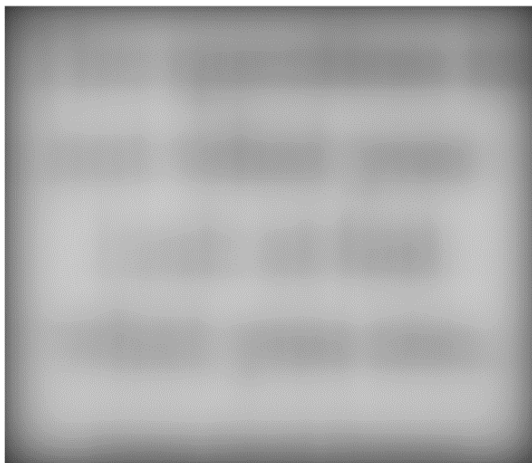
Figure 1

What do you see in Figure 1? Some may see a square with smaller squares of varying shades of grey, others may see one big square depending on their vision. However, this is a matrix with pixel intensities. As you may store this information as what you see, a large square, the computer stores this as a matrix with a black and white intensity between 0-255. When your brain processes the information coming from your eyes, you can see a greater variety of color intensities than the computer stores. It's just that the difference between the true intensity and the one that the computer stores is so small that the human eye has trouble telling the difference between them.

In order to deblur an image, we must first create a formula to represent the blurring process. The general model for a blurred image can be given as follows: $\mathbf{Ax} + \mathbf{e} = \mathbf{b}$. in this equation: \mathbf{A} is a matrix that includes the blur, \mathbf{x} is the exact image, \mathbf{e} is the error otherwise known as the noise, and lastly \mathbf{b} is the blurred photo. The lowercase notation in our formula means that \mathbf{x} , \mathbf{e} , and \mathbf{b} are all vectors, rather than uppercase which denotes a matrix. Noise can be caused by various things including discretization, mechanical, and human error. Discretization error occurs when a continuous variable is stored as a finite variable, whereas mechanical error comes from error of the device capturing the image, and human error comes from the individual taking the photo. Unfortunately, we are unable to estimate this noise, so we are unable to recover the original image with our mathematical model. The general formula we are trying to solve for can be given as follows: $\mathbf{x} = \mathbf{A}^{-1}(\mathbf{b} - \mathbf{e}) = \mathbf{A}^{-1}\mathbf{b} - \mathbf{A}^{-1}\mathbf{e}$. In this case we know $\mathbf{A}^{-1}\mathbf{b}$, but we don't know $\mathbf{A}^{-1}\mathbf{e}$ or \mathbf{x} . A naïve approach is to assume that \mathbf{x} is approximately $\mathbf{A}^{-1}\mathbf{b}$ which occurs only when the inverted noise, $\mathbf{A}^{-1}\mathbf{e}$, is small relative to the size of $\mathbf{A}^{-1}\mathbf{b}$.

The following two examples use the naïve approach to attempt to deblur an image:

Original Photo



Results

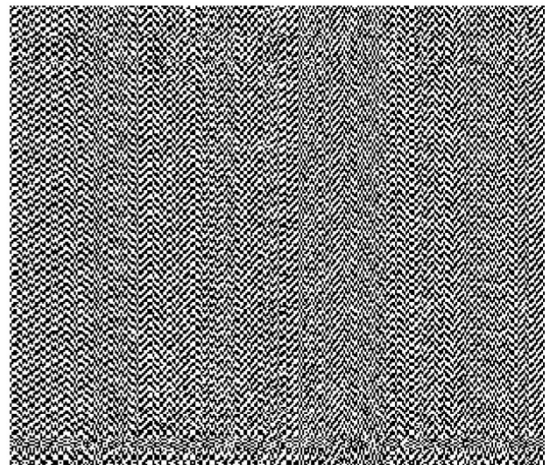
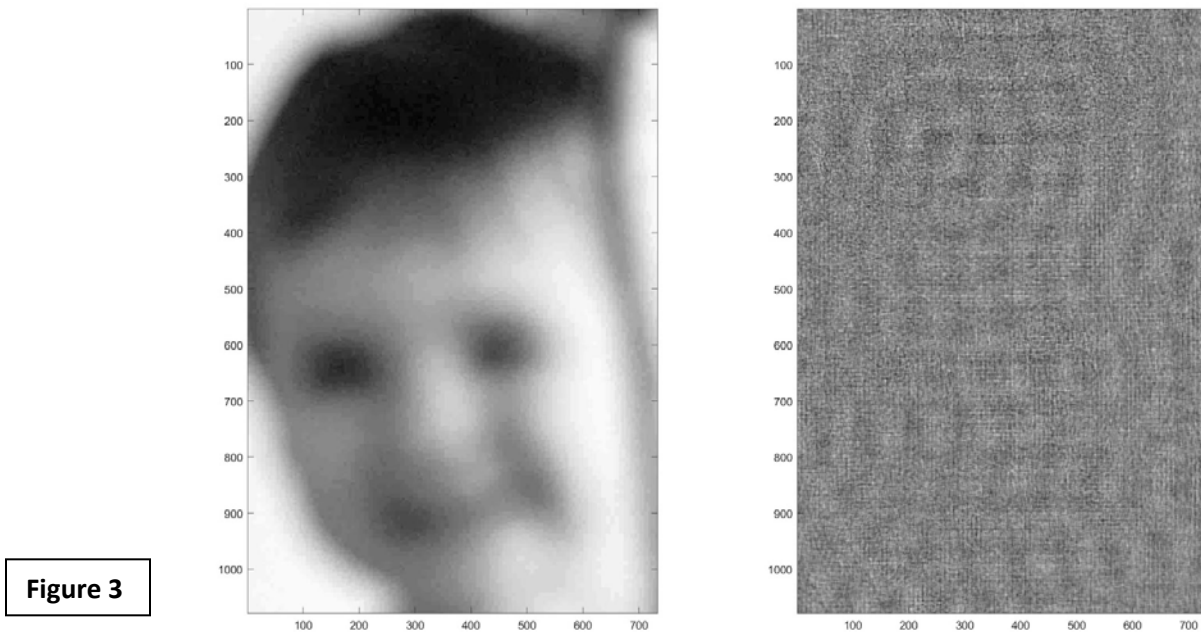


Figure 2



In Figures 2 and 3, we are attempting to reconstruct the original image using the naïve approach. However, the inverted noise ($\mathbf{A}^{-1}\mathbf{e}$) is much larger than the reconstructed image ($\mathbf{A}^{-1}\mathbf{b}$) and dominates both photos. The image in Figure 2 is taken from [1].

Deblurring using a general linear model

Assuming the blur is linear, we have tools to reduce the effect of the inverted noise while simultaneously not changing the value of $\mathbf{A}^{-1}\mathbf{b}$ very much. Two popular image deblurring methods that will be considered in this thesis are the singular value decomposition (SVD) and the discrete cosine transform (DCT). Both the SVD and DCT require user input to choose the value of a parameter which determines how much the inverted noise is reduced, but there is a trade-off. As you reduce (or dampen) the inverted noise, the more information from the exact image is being lost.

Both the singular value decomposition and the discrete cosine transform decompose the image according to the frequency of the signal. These decompositions allow us to dampen the inverted noise in order to make the image the highest quality possible.

Before we go into the mathematics of both the singular value decomposition and the discrete cosine transform, we must talk about the point spread function and boundary conditions.

Point Spread Functions (PSF) and Boundary Conditions

A point spread function describes the way in which the blur occurs. Although there are many blurs including horizontal blur, atmospheric turbulence blur, and Moffat blur, we are assuming an out-of-focus blur. An out-of-focus blur models the effect of the camera lens not being focused properly. In Figure 4, the white area corresponds to the pixels whose values are being averaged together in order to achieve the blur. Keep in mind that this blur is being applied every pixel in the image, so information outside of the frame of the image is being used to blur the image.

Thus, we must explore what are called boundary conditions. Boundary conditions are the assumptions about what information is contained outside of the frame of the exact image. To attempt to reproduce the exact image many blurring models account for this loss of information. The main boundary conditions are periodic, zero, and reflexive.

Out-of-focus blur

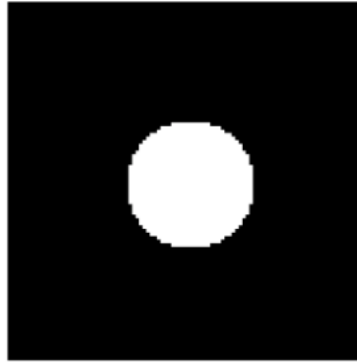


Figure 4

The image in Figure 4 is taken from [1].

Boundary Conditions

We will consider three boundary conditions: periodic, zero, and reflexive. In each of the following examples we assume our exact image is the part of the matrix given in red and the numbers in black is the information stored outside of the frame.

Periodic Boundary Condition

For a periodic boundary condition, we assume that the matrix repeats itself outside of the exact

image:

1	2	3	1	2	3	1	2	3
4	5	6	4	5	6	4	5	6
7	8	9	7	8	9	7	8	9
1	2	3	1	2	3	1	2	3
4	5	6	4	5	6	4	5	6
7	8	9	7	8	9	7	8	9
1	2	3	1	2	3	1	2	3
4	5	6	4	5	6	4	5	6
7	8	9	7	8	9	7	8	9

Zero-Boundary Condition

For a zero-boundary condition, we assume that the pixels stored outside of our exact image are black, which corresponds to a grayscale intensity of 0:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 5 & 6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7 & 8 & 9 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Reflexive Boundary Condition

For a reflexive boundary condition, we assume that a mirror image of the exact image appears outside of the frame:

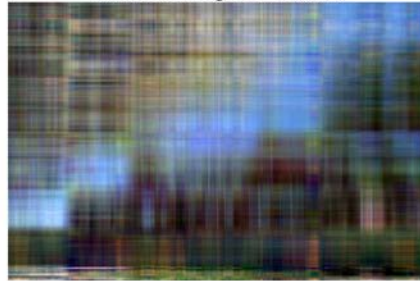
$$\begin{bmatrix} 9 & 8 & 7 & 7 & 8 & 9 & \square & 8 & 7 \\ 6 & 5 & 4 & 4 & 5 & 6 & 6 & 5 & 4 \\ 3 & 2 & 1 & 1 & 2 & 3 & 3 & 2 & 1 \\ 3 & 2 & 1 & 1 & 2 & 3 & 3 & 2 & 1 \\ 6 & 5 & 4 & 4 & 5 & 6 & 6 & 5 & 4 \\ 9 & 8 & 7 & 7 & 8 & 9 & 9 & 8 & 7 \\ 9 & 8 & 7 & 7 & 8 & 9 & 9 & 8 & 7 \\ 6 & 5 & 4 & 4 & 5 & 6 & 6 & 5 & 4 \\ 3 & 2 & 1 & 1 & 2 & 3 & 3 & 2 & 1 \end{bmatrix}$$

How the SVD works

The Singular Value Decomposition (SVD) is a matrix decomposition which allows us to separate our image into a sum of images. The relationship between the size of the singular value and its contribution to the final image is best described visually.



Number of Singular Values : 5



Number of Singular Values : 50



Number of Singular Values : 100



Number of Singular Values : 300



The image on the left is the exact image, whereas the image on the right represents the reconstructed image with a certain number of singular values. As you can see, the larger singular

values have a lower frequency and contribute lots of information to the image, whereas the singular values close to zero have a higher frequency and contribute very little to the overall image. This is an important concept of this thesis since it shows us that the higher-frequency singular values contain the granular detail, but also contribute the most to the inverted noise that we are trying to dampen. The SVD is extremely useful since it identifies portions of the image signal which should be dampened or eliminated in order to reduce inverted noise. In most cases, the SVD assumes a zero-boundary condition.

SVD Background

Given a $m \times n$ matrix \mathbf{A} , the formula for the Singular Value Decomposition can be given as follows: $\mathbf{A} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T$. In this model we know that \mathbf{U} and \mathbf{V} are orthogonal matrices and $\mathbf{\Sigma}$ is a diagonal matrix whose elements σ_i are nonnegative and appear in nonincreasing order. \mathbf{A} is equal to sum of rank one matrices.

$$\mathbf{A} = \sigma_1 * \mathbf{u}_1 * \mathbf{v}_1^T + \sigma_2 * \mathbf{u}_2 * \mathbf{v}_2^T + \dots + \sigma_n * \mathbf{u}_n * \mathbf{v}_n^T$$

To calculate a singular value decomposition, first compute $\mathbf{A}^T \mathbf{A}$. The square root of the eigenvalues of $\mathbf{A}^T \mathbf{A}$ are the singular values of \mathbf{A} , which are denoted by σ_i . The columns \mathbf{u}_i of the matrix \mathbf{U} are the normalized eigenvectors of $\mathbf{A}^T \mathbf{A}$, and the columns of \mathbf{V} are given by $\frac{\mathbf{A} \mathbf{u}_i}{\sigma_i}$. An example of computing the singular value decomposition given a matrix is as follows:

(Example taken from [2])

$$\mathbf{A} = \begin{bmatrix} 2 & -1 \\ 2 & 2 \end{bmatrix}$$

$$A^T = \begin{bmatrix} 2 & 2 \\ -1 & 2 \end{bmatrix}$$

$$A^T A = \begin{bmatrix} 2 & 2 \\ -1 & 2 \end{bmatrix} \begin{bmatrix} 2 & -1 \\ 2 & 2 \end{bmatrix} = \begin{bmatrix} 8 & 2 \\ 2 & 5 \end{bmatrix}$$

Compute the determinant of $A^T A - \lambda I = 0$

$$A^T A - \lambda I = 0 \rightarrow \begin{vmatrix} 8 - \lambda & 2 \\ 2 & 5 - \lambda \end{vmatrix} = 0$$

$$A^T A - \lambda I = (8 - \lambda)(5 - \lambda) - (2)(2)$$

$$40 - 13\lambda + \lambda^2 - 4 = 0$$

$$36 - 13\lambda + \lambda^2 = 0$$

$$\lambda_1 = 9$$

$$\lambda_2 = 4$$

$$\sigma_1 = \sqrt{9} = 3$$

$$\sigma_2 = \sqrt{4} = 2$$

$$\Sigma = \begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix}$$

$$\Sigma^{-1} = \begin{bmatrix} \frac{1}{3} & 0 \\ 0 & \frac{1}{2} \end{bmatrix}$$

$$A^T A - \lambda_1 I = 0 \rightarrow \begin{bmatrix} 8 - 9 & 2 \\ 2 & 5 - 9 \end{bmatrix} = \begin{bmatrix} -1 & 2 \\ 2 & -4 \end{bmatrix} \Rightarrow \begin{matrix} x_1 = 2 \\ x_2 = 1 \end{matrix}$$

Divide by its length/ magnitude:

$$L = \sqrt{2^2 + 1^2} = \sqrt{5}$$

$$v_1 = \begin{bmatrix} \frac{2}{L} \\ \frac{1}{L} \end{bmatrix} = \begin{bmatrix} \frac{2}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} \end{bmatrix}$$

Compute u_1 as $\frac{1}{\sigma_1} A v_1$

$$u_1 = \frac{1}{3} \begin{bmatrix} 2 & -1 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} \frac{2}{\sqrt{5}} \\ 1 \\ \frac{1}{\sqrt{5}} \end{bmatrix}$$

$$u_1 = \frac{1}{3} \begin{bmatrix} 3 \\ \frac{3}{\sqrt{5}} \\ 6 \\ \frac{6}{\sqrt{5}} \end{bmatrix}$$

$$u_1 = \begin{bmatrix} 1 \\ \frac{1}{\sqrt{5}} \\ 2 \\ \frac{2}{\sqrt{5}} \end{bmatrix}$$

$$A^T A - \lambda_2 I = 0 \rightarrow \begin{bmatrix} 8 & -4 & 2 \\ & 2 & -4 \end{bmatrix} = \begin{bmatrix} 4 & 2 \\ 2 & 1 \end{bmatrix} \Rightarrow \begin{matrix} x_1 = -1 \\ x_2 = 2 \end{matrix}$$

Divide by its length/ magnitude

$$L = \sqrt{-1^2 + 2^2} = \sqrt{5}$$

$$v_2 = \begin{bmatrix} \frac{-1}{L} \\ \frac{2}{L} \\ \frac{2}{L} \end{bmatrix} = \begin{bmatrix} \frac{-1}{\sqrt{5}} \\ \frac{2}{\sqrt{5}} \\ \frac{2}{\sqrt{5}} \end{bmatrix}$$

Compute u_2 as $\frac{1}{\sigma_2} A v_2$

$$u_2 = \frac{1}{2} \begin{bmatrix} 2 & -1 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} \frac{-1}{\sqrt{5}} \\ \frac{2}{\sqrt{5}} \\ \frac{2}{\sqrt{5}} \end{bmatrix}$$

$$u_2 = \frac{1}{2} \begin{bmatrix} -4 \\ \frac{-4}{\sqrt{5}} \\ 2 \\ \frac{2}{\sqrt{5}} \end{bmatrix}$$

$$\boldsymbol{u}_2 = \begin{bmatrix} -2 \\ \frac{1}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} \end{bmatrix}$$

$$\mathbf{U} = [\boldsymbol{u}_1 \quad \boldsymbol{u}_2] = \begin{bmatrix} \frac{1}{\sqrt{5}} & \frac{-2}{\sqrt{5}} \\ \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} & \frac{1}{\sqrt{5}} \end{bmatrix}$$

$$\mathbf{V} = [\boldsymbol{v}_1 \quad \boldsymbol{v}_2] = \begin{bmatrix} \frac{2}{\sqrt{5}} & \frac{-1}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} & \frac{2}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} & \frac{1}{\sqrt{5}} \end{bmatrix}$$

$$\mathbf{V}^T = \begin{bmatrix} \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{5}} \\ \frac{-1}{\sqrt{5}} & \frac{2}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} & \frac{1}{\sqrt{5}} \end{bmatrix}$$

$$\mathbf{A} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T$$

$$\mathbf{A} = \begin{bmatrix} \frac{1}{\sqrt{5}} & \frac{-2}{\sqrt{5}} \\ \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} & \frac{1}{\sqrt{5}} \end{bmatrix} \begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{5}} \\ \frac{-1}{\sqrt{5}} & \frac{2}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} & \frac{1}{\sqrt{5}} \end{bmatrix}$$

$$\mathbf{A} = 3 \begin{bmatrix} \frac{1}{\sqrt{5}} \\ \frac{2}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} \end{bmatrix} \begin{bmatrix} \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{5}} \end{bmatrix} + 2 \begin{bmatrix} \frac{-2}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} \end{bmatrix} \begin{bmatrix} \frac{-1}{\sqrt{5}} & \frac{2}{\sqrt{5}} \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} \frac{6}{5} & \frac{3}{5} \\ \frac{12}{5} & \frac{6}{5} \\ \frac{5}{5} & \frac{5}{5} \end{bmatrix} + \begin{bmatrix} \frac{4}{5} & \frac{-8}{5} \\ \frac{-2}{5} & \frac{4}{5} \\ \frac{5}{5} & \frac{5}{5} \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} 2 & -1 \\ 2 & 2 \end{bmatrix}$$

Multiplying out the matrix will give us the exact matrix we started with. In a matrix that has many more singular values, the smaller singular values contribute much less to the overall image and can be removed. By removing some of the smaller singular values, we are dampening the inverted noise.

Moore-Penrose inverse

The calculation of the inverse of the singular value decomposition of \mathbf{A} requires calculating the Moore-Penrose inverse. The Moore-Penrose inverse is a pseudo-inverse which behaves similarly to an inverse but does not have all the properties of an inverse. We will use the Moore-Penrose inverse to approximate the deblurred image $\mathbf{A}^{-1}\mathbf{b}$. In our case, $\mathbf{\Sigma}$ is square so we do not need to pad the matrix. The general shape of a square (n x n) $\mathbf{\Sigma}$ is as follows:

$$\begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \sigma_n \end{bmatrix}$$

When taking the inverse of a square (n x n) $\mathbf{\Sigma}$ is given as follows:

$$\begin{bmatrix} \frac{1}{\sigma_1} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \frac{1}{\sigma_n} \end{bmatrix}$$

Since this matrix is square (n x n) the inverse is the real inverse, however, if the inverse were rectangular (m x n) it would be padded with row(s) or column(s) of zeros and would be a pseudo-inverse.

The formula for the Moore-Penrose inverse of the matrix

$$\mathbf{A} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T = \sum_{i=1}^N \sigma_i \mathbf{u}_i \mathbf{v}_i^T$$

is given as follows:

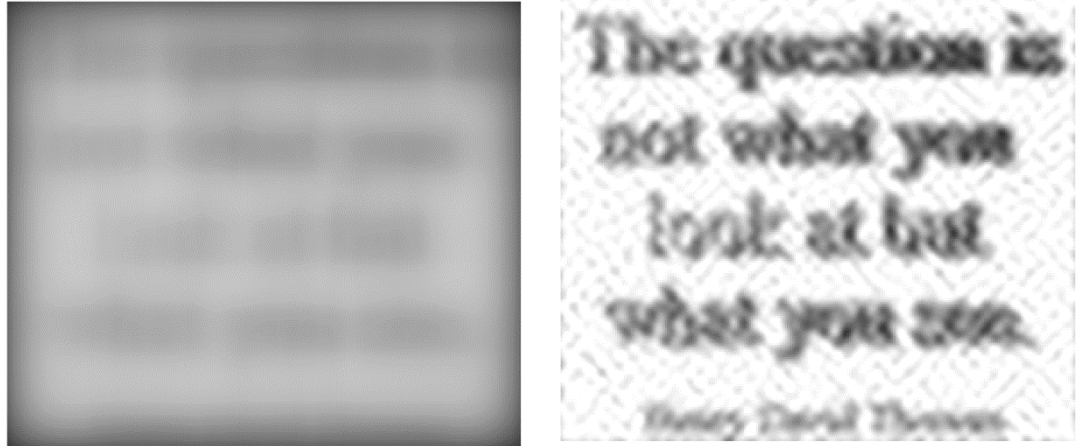
$$\mathbf{A}^+ = \mathbf{V} \mathbf{\Sigma}^+ \mathbf{U}^T = \sum_{i=1}^N \frac{1}{\sigma_i} \mathbf{v}_i \mathbf{u}_i^T.$$

Since our general model is given by $\mathbf{Ax} + \mathbf{e} = \mathbf{b}$ we can attempt to reduce inverted noise by removing the singular values that make $\mathbf{A}^+\mathbf{e}$ large. We assume $\mathbf{X} = \mathbf{A}^+\mathbf{b}$ and we ignore the error term because we don't know what $\mathbf{A}^+\mathbf{e}$ is. We hope through this process we have made the inverted noise small. However, we are making a tradeoff of a lower error for a lower quality reconstructed image. We can keep a higher image quality, but the noise will ultimately dominate as seen in Figure 3, or we can sacrifice image quality for dampened noise. This can be described by removing the higher-frequency pieces but keeping the lower-frequency pieces.

Application to Deblurring

The singular value decomposition can be applied to image deblurring as a tool to dampen the inverted noise. As you can see in Figure 5, by applying the SVD to our image we are able to recover much of the information that was lost in the blurring process, although we lost some of the detail that was embedded in the portion of the signal corresponding to small singular values.

Figure 5



Discrete Cosine Transform Background

The Discrete Cosine Transform (DCT) is another matrix decomposition which allows us to decompose the image into a sum of images. The DCT is useful since with the individual sum of images we can dampen the inverted noise. The DCT is most useful when assuming a reflexive boundary condition. The main difference between the DCT and the SVD is the computational efficiency when computing \mathbf{A} . However, this algorithm is only computationally efficient when reflexive boundary conditions are used.

Definition

The Discrete Cosine Transform (DCT) is defined as:

$$\mathbf{A} = \mathbf{C}^T \mathbf{\Lambda} \mathbf{C}$$

$\mathbf{C}^T \mathbf{\Lambda} \mathbf{C}$ can be written similarly to the SVD, where \mathbf{A} is equal to sum of rank one matrices.

$$\mathbf{A} = \lambda_1 * \mathbf{c}_1 * \mathbf{c}_1^T + \lambda_2 * \mathbf{c}_2 * \mathbf{c}_2^T + \dots + \lambda_n * \mathbf{c}_n * \mathbf{c}_n^T$$

In this formula, matrix \mathbf{C} is an orthogonal two-dimensional Discrete Cosine Transform matrix (see [1]). Unlike in the singular value decomposition we only need to compute $\mathbf{\Lambda}$ which is computationally efficient when $\mathbf{\Lambda}$ is diagonal. If blurring matrix \mathbf{A} has reflexive boundary conditions, $\mathbf{\Lambda}$ is diagonal, then the diagonal entries can be found using the following formula:

$$\lambda_i = \frac{[\mathbf{C}\mathbf{a}_1]_i}{c_{i1}}$$

The benefit of the DCT is that rather than needing to calculate three different matrices, like in the SVD with \mathbf{U} , $\mathbf{\Sigma}$, and \mathbf{V} , we only need to calculate $\mathbf{\Lambda}$. If $\mathbf{\Lambda}$ is diagonal, the eigenvalues of $\mathbf{\Lambda}$ are exactly the diagonal elements of the matrix. Thus, calculating the eigenvalues tells us exactly the values of $\mathbf{\Lambda}$. However, the point is to find the eigenvalues so that we can write the matrix \mathbf{A} as the sum of rank one matrices, just like in the SVD case. The eigenvalues will take the place of the singular values. In this equation \mathbf{a}_1 is the first column of \mathbf{A} and c_{i1} is an element of \mathbf{C} , see [1]. The matrix \mathbf{C} is determined by the size of \mathbf{A} , and not the entries of \mathbf{A} . The Discrete Cosine Transform is used for the popular image compression format known as JPEG.

Example of the Discrete Cosine Transform

$$\mathbf{A} = \begin{bmatrix} 2 & 4 & 6 \\ 4 & 4 & 4 \\ 6 & 4 & 2 \end{bmatrix}$$

Notice that the matrix \mathbf{A} is doubly reflexive, so the discrete cosine transform will be a diagonal matrix. The 3x3 discrete cosine transform matrix is approximately equal to

$$\mathbf{C} = \begin{bmatrix} 0.5774 & 0.5774 & 0.5774 \\ 0.7071 & 0 & -0.7071 \\ 0.4082 & -0.8165 & 0.4082 \end{bmatrix},$$

and

$$\mathbf{A} = \mathbf{C}^T \mathbf{\Lambda} \mathbf{C} = \mathbf{C}^T \begin{bmatrix} 12 & 0 & 0 \\ 0 & -4 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{C}.$$

We can see that the diagonal entries of $\mathbf{\Lambda}$ can be computed by first calculating

$$\mathbf{C}\mathbf{a}_1 = \begin{bmatrix} 0.5774 & 0.5774 & 0.5774 \\ 0.7071 & 0 & -0.7071 \\ 0.4082 & -0.8165 & 0.4082 \end{bmatrix} \begin{bmatrix} 2 \\ 4 \\ 6 \end{bmatrix} = \begin{bmatrix} 6.9282 \\ -2.8284 \\ 0 \end{bmatrix}$$

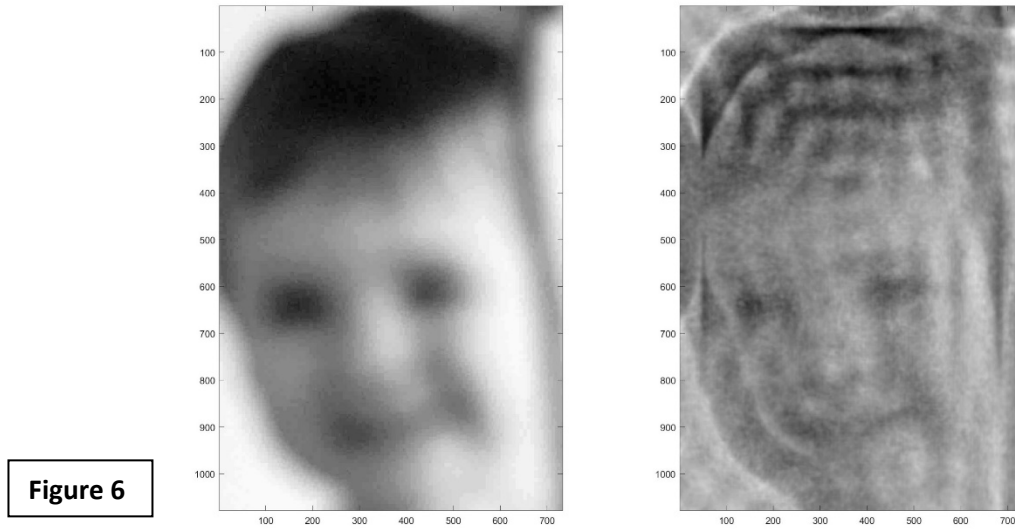
and then dividing each entry by the corresponding entry of the first column of \mathbf{C} :

$$\lambda_1 = \frac{[\mathbf{C}\mathbf{a}_1]_1}{c_{11}} = \frac{6.9282}{0.5774} = 12, \lambda_2 = \frac{[\mathbf{C}\mathbf{a}_1]_2}{c_{21}} = \frac{-2.8284}{0.7071} = -4, \lambda_3 = \frac{[\mathbf{C}\mathbf{a}_1]_3}{c_{31}} = \frac{0}{0.4082} = 0.$$

For even moderately sized images, the number of floating point operations necessary to calculate the eigenvalues of $\mathbf{\Lambda}$ in this fashion is far less than the number of operations necessary to calculate the eigenvalues using the singular value decomposition. For this reason, the discrete cosine transform is the preferred matrix decomposition when using reflexive boundary conditions. For other boundary conditions, the entries of $\mathbf{\Lambda}$ cannot be computed in the same way.

Application to Deblurring

The Discrete Cosine Transform can be applied to image deblurring as a tool to dampen the inverted noise. As you can see in Figure 6 by applying the DCT to our image we are able to recover much of the information that was lost in the blurring process. Although we lost some of the detail that came with the inverted noise, we recovered most of the information that was lost during the blurring process.



Spectral Filtering Introduction

Spectral filtering is a technique which reduces the contribution of small singular values (or small eigenvalues) and leaves large singular values relatively unchanged. Small singular values are multiplied by a value of $\varphi(i)$ that is close to zero, whereas large singular values are multiplied by a value of $\varphi(i)$ that is close to one:

$$\mathbf{A} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T = \sum_{i=1}^N \varphi(i) \sigma_i \mathbf{u}_i \mathbf{v}_i^T$$

In our project we used multiple spectral filtering techniques including the Tikhonov method and the cutoff method to reduce the inverted noise.

Tikhonov Method

$$\varphi(i) = \frac{\sigma_i^2}{\sigma_i^2 + \alpha^2}$$

If α is much greater than the singular value, then $\varphi(i)$ is around 0. When we multiply $\varphi(i)$ in our SVD formula we get a value close to 0, which suppresses the inverted noise from the small singular values.

If α is much smaller than the singular value, then $\varphi(i)$ is around 1. When we multiply $\varphi(i)$ in our SVD formula we get a value close to its original value. We aren't changing the large singular values because it contributes a lot to the overall photo and there is not a lot of inverted noise.

Unlike the cutoff method, we do not zero out the contribution of the smaller singular values, rather we reduce its contribution to the overall image. To do this, we multiply its value by a number close to, but not exactly zero.

(Truncated SVD) Singular Value or Eigenvalue Cutoff Methods

The cutoff method uses the same formula as the Tikhonov method:

$$\mathbf{A} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T = \sum_{i=1}^N \varphi(i) \sigma_i \mathbf{u}_i \mathbf{v}_i^T$$

However, instead of getting values close to 0 and close to 1, the cutoff method chooses a k value where:

$$\varphi(i) = \begin{cases} 1, & \text{if } i \leq k \\ 0, & \text{if } i > k \end{cases}$$

Unlike the Tikhonov method, the cutoff method is all or nothing and the values at a certain point either contribute or don't to the overall photo. A similar technique can be used on decompositions from the discrete cosine transform.

Conclusion

As we approach digital image deblurring, we must think of a photo as a matrix of numbers corresponding to different color intensities. The image deblurring problem can be approximated by a linear model $\mathbf{Ax} + \mathbf{e} = \mathbf{b}$. In this equation, \mathbf{A} is a matrix that includes the blur, \mathbf{x} is the exact image, \mathbf{e} is the error otherwise known as the noise, and lastly \mathbf{b} is the blurred photo. We must make an assumption about the structure of the blurring matrix \mathbf{A} , which describes the type of blur that is occurring. We must also make an assumption about the type of boundary conditions that are appropriate for the image. A naïve approach to deblurring is dominated by the inverse noise, which cannot be estimated. In this thesis, two different matrix decompositions known as the singular value decomposition (SVD) and the discrete cosine transform (DCT) are used to decompose the matrix \mathbf{A} and identify portions of its spectrum that contribute most to the inverted noise. Depending on the circumstances, one matrix decomposition might be more useful than the other. Using spectral filtering techniques, the inverted noise $\mathbf{A}^+\mathbf{e}$ is suppressed. However, since spectral filtering techniques alter the value of $\mathbf{A}^+\mathbf{b}$ and we cannot estimate \mathbf{e} , it is impossible to recover the exact image.

References

- [1] P. C. Hansen, J. G. Nagy, D. P. O’Leary; *Deblurring images matrices, spectra, and filtering*, SIAM, Philadelphia (2006)
- [2] D. C. Lay, S. R. Lay, J. J. McDonald; *Linear Algebra and its Applications FIFTH EDITION*, Pearson, London (2015)